Syukri Shukor
CSCI E-101
Managing Data Exercise 1

CSCI E-101: Managing Data Exercise 1

Syukri Shukor

Challenges:

I. –secure-file-priv

Before uploading the full dataset, I created a smaller test.csv file containing the first 8 rows of the dataset.

The challenge in uploading my data is that running the following chunk

```
SQLCMD = "LOAD DATA INFILE 'C:/Users/sykri/PycharmProjects/CSCI E-101 Week
4/nj_state_teachers_salaries_test.csv'\
         INTO TABLE nj_state_teachers_salaries.nj_state_teachers_salaries \
         FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\n' IGNORE 1 ROWS"
mycursor.execute(SQLCMD)
mydb.commit()
```

throws an error that MySQL server is running with the --secure-file-priv option.

To solve this, in MySQL Workbench, running 'SHOW VARIABLES LIKE "secure_file_priv"' prints the following: 'secure_file_priv', 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\'.

This path leads to a folder where MySQL allows data uploads. Moving the .csv file to this folder and updating the previous chunk to the following proper link led to successful data upload.

```
SQLCMD = "LOAD DATA LOCAL INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/nj_state_teachers_salaries_test.csv'\
         INTO TABLE nj_state_teachers_salaries.nj_state_teachers_salaries \
         FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\n' IGNORE 1 ROWS"

mycursor.execute(SQLCMD)
mydb.commit()
```

II. Handling various data types

Another challenge was in defining the data types for each column. First, the INT data type used to address the last 3 columns could not accept NULL values. Second, certain string fields could potentially be longer than what VARCHAR(1 – 255) could handle. Thus, all columns were set as TEXT. This implementation was chosen since space and time processing is not a priority in solving this problem.

III. Addressing commas within strings

The test.csv file was updated with another row (9th row). This new entry's primary job contains the string 'School Librarian, Media Specialist Assoc'.

Re-running the above chunk in **b.,** which worked in the previous test case**,** with this new test case threw an error:

```
mysql.connector.errors.DatabaseError: 1265 (01000): Data truncated for column 'fte' at row 8
```

Closer inspection showed that the 'fte' column comes after 'primary_job'. Within the 'primary_job' column, the MySQL-connector interpreted the ',' in 'School Librarian, Media Specialist Assoc' as 2 separate column objects 'School Librarian' adnd 'Media Specialist Assoc', hence deviating from the

defined table structure. To fix this interpreter issue, we include an OPTIONALLY ENCLOSED BY '\"' statement.

```
SQLCMD = "LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/nj_state_teachers_salaries.csv'\
        INTO TABLE nj_state_teachers_salaries.nj_state_teachers_salaries \
        FIELDS TERMINATED BY ','\
        OPTIONALLY ENCLOSED BY '\"'\
        LINES TERMINATED BY '\n' IGNORE 1 ROWS"
```

This OPTIONALLY ENCLOSED BY '\"' forces the FIELDS TERMINATED BY line to ignore ',' within strings, thus, reporting 'School Librarian, Media Specialist Assoc.' as a single string.

After addressing the challenges in **a.** and **b.**, the code was applied on the full dataset. The uploaded table was then exported from MySQL Workbench and compared with the original dataset for accuracy. All rows were imported, all columns are present, and no changes in order of rows nor columns were observed.